ITCS 209 Object Oriented Programming	Name:	Lab Score	Challenge Bonus	
	ID:			
	Section:			

Lab05: Arrays and ArrayLists

Since this lab aims to test specific aspects of Java/OOP, you can only use the following mechanisms available in Java: **loops, arrays, List, ArrayList, primitive data types, Double, String, methods, and classes**. You may NOT use Math and other computation libraries, except for Math.sqrt() and Math.pow(). All the computation must be implemented from scratch using loops, arrays, and/or Lists. You may not use other container libraries (e.g., Set, Map, etc.) other than ArrayList. You are also NOT allowed to use third-party Java libraries. Your code should be able to compile and run without having to install external jar files.

In this lab, you have to **choose one** of the following assignment.

Assignment #1 Bank Application

You are provided with the **`BankTester`** class and **`BankAccount`** class. Do not modify these classes. Your task is to create **Bank** class and implement the following attributes and methods:

- Attributes or instance variables:
  - A bank's name (String), a list of accounts (using ArrayList to store BankAccount objects)
- Constructor method:
  - Take String name as an argument and assign to the bank's name and construct arraylist of bank accounts
- Instance methods:
  - getName() which
    - Takes no parameters and returns bank's name
  - addAccount(*BankAccount* bankAccount)
    - Takes 1 parameter which is a *BankAccount* object and adds into the arraylist
  - getTotalBalance() which
    - Takes no parameters and returns *totalbalance* in double data type
  - countBalanceAtLeast(double atLeast) which
    - Takes 1 parameter which is the balance required to count an account
    - Returns *the number of accounts* having at least the given balance
  - find(int accountNumber) which
    - Takes 1 parameter which is the account number to find
    - Returns a *BankAccount* corresponding to the accountNumber, or null if there is no such account.
  - getMax () which
    - Takes no parameters
    - Returns a *BankAccount* which the highest balance, or null if the bank has no accounts. If there is more than one BankAccount with the max balance, return one of them.
  - getMin () which
    - Takes no parameters
    - Returns a *BankAccount* which the smallest balance, or null if the bank has no accounts. If there is more than one BankAccount with the minimum balance, return one of them.

Expected Output:

```
Bank Name: +POSITIVE+
Total balance: 45000.00
Account number with having balance at least 12000: 2
Balance of matching account: 15000.0
```

Account number with the smallest balance: 1015

Account number with the highest balance: 1001

Cannot find account 1730

****

Bank Name: +++BIG POSITIVE+++

Total balance: 45000.00

Account number with having balance at least 5000: 26

Balance of matching account: 8048.61

Account number with the smallest balance: 6900

Account number with the highest balance: 3185

Cannot find account 4242

Challenge Assginment #1

Implement **`List<Integer> findDuplicate()`** in Bank that returns the list of bank accounts' number that are later found duplicated, so the back staff can further investigate fraudulent activities. Your code can only use lists and arrays. Therefore, Set and Map interfaces are not allowed at this point (Don't worry, you will get to use Set and Map soon).

Expected Output:

********** BONUS **********

Found duplicate accounts: [BankAccount [acctnumber=6900, balance=1825.35], BankAccount [acctnumber=6900, balance=60.25], BankAccount [acctnumber=6900, balance=71.25], BankAccount [acctnumber=1969, balance=1043.59], BankAccount [acctnumber=1969, balance=1044.59]]

Assignment #2 Song Application

You are provided with the **SongApp** class and **Song** class. Do not modify these classes. The main method is already provided in SongApp class. Your task is to Create a **`Playlist`** class and implement the following attributes and methods:

- Attributes or instance variables:
  - A playlist's name (String), a list of songs (using ArrayList to store Song objects)
- Constructor method:
  - Take String name as an argument and assign to the playlist's name and construct arraylist of songs
- Instance methods:
  - `boolean addSong(Song song)`
    - to add a new song into the playlist. Note that the song with the same title can be added only once.
    - returns *true* if the song is a new song and can be added to the playlist. If the song already exists, do nothing and return *false*.
  - `boolean addSongAtIndex(Song song, int index)`
    - to add a new song into the playlist at a given position.
    - returns *true* if the song is a new song and can be added to the playlist. If the song already exists, do nothing and return *false*. If the index is invalid, you must display an error message (see expected output) and return false. Note that the index in the playlist starts at '0'.
  - `boolean removeSongByIndex(int index)`
    - to remove an existing song from the playlist at a given index
    - returns true if there was a song at the given index, otherwise display an error message and return false
  - `Song removeSongByTitle(String title)`
    - to remove an existing song from the playlist by looking at the song title
    - returns a *Song* corresponding to the title, or null if there is no such song.
  - `double getPlaylistDuration()`

        - retuns the total duration of the playlist Each minute only have 60 seconds. So 3.30 minute + 3.40 minutes = 7.10 minutes (not 6.70 minutes)
    - ArrayList<Song> getTooLongSongs (double minute)
        - returns all songs in the playlist that is too long based on user's preference in minutes.
    - `void showPlaylist()`
        - to print the playlist on the console in the following format

playlist name [0] *title* (*duration*) [1] *title* (*duration*) ….

Note that: For add and remove methods, you suppose to show (print out) an error message when the given index or title does not existing in your current playlist. Pleas see the expected output for some example messages.
Here is the expected output after you run the SongApp.java class.

```
Welcome to SongAPP


Add songs -------------------------
My Favorite Songs Playlist
[0] End Game (4.11)
[1] Perfect (4.21)
[2] Anywhere (3.35)
[3] How long (3.30)

Remove songs ----------------------
My Favorite Songs Playlist
[0] Perfect (4.21)
[1] Anywhere (3.35)

Check error ------------------------
Add duplicate song: Perfect already in the playlist.
Add song at invalid index (at most index now is 2): Invalid Index
Remove not found song: Not found
Remove song at invalid index: Invalid Index
```

Challenge Assginment #2

- `void moveUp(int position)`
    - to rearrange the position of songs in the playlist by moving the song at the given index up by one position.
- `void moveDown(int position)`
    - to rearrange the position of songs in the playlist by moving the song at the given index down by one position.

Expected Output:

```
********** BONUS **********
Bonus Playlist
[0] End Game (4.11)
[1] Perfect (4.21)
[2] Anywhere (3.35)
[3] How long (3.30)
```

Rearrange songs ----------

Bonus Playlist

[0] Perfect (4.21)

[1] End Game (4.11)

[2] How long (3.30)

[3] Anywhere (3.35)

// Array Practice (Just for fun - no points)

Write a Java program to check palindrome. A palindrome is a word, phrase, number, or other sequence of characters which can be read the same way from backward or forward. For example, "madam", "mom", "abcba", and "123321" are single word palindrome. "Don't nod.", "Top spot" and "No lemon, no melon" are multiple words palindrome. *(Punctuation and spaces between the words or letter is allowed.)* "Java", and "Array" are NOT palindrome. The longest palindrome in Oxford dictionary is "tattarrattat".

a. You may write a `Class` or simply have everything in the `main(String[] args)` method.
b. Your program has to accept input `String` from users and check whether the input `String` is a palindrome or not. An example output is shown in the box below:

```
Enter a word or phrase to check if it is a palindrome: tattarrattat
The input word "tattarrattat" is a palindrome

Enter a word or phrase to check if it is a palindrome: I love java
The input phrase "I love Java" is not a palindrome
```
